High efficient robotic de-palletizing system for the non-flat ceramic industry

J. Norberto Pires
Mechanical Engineering Department
University of Coimbra
norberto@robotics.dem.uc.pt

Sergio Paulo
Roca Cerâmica e Comércio, SA - PORTUGAL
sergiopaulo@roca.net

I. Abstract and Introduction

In this paper we describe a system designed to be used on a shop floor of a factory that produces non-flat ceramic products, where an extensive mixture of human and automatic labour is present. Today manufacturing setups rely increasingly on technology. It is common to have all sources of equipment on the shop floor, commanded by industrial PCs or PLCs connected by an industrial network to other factory resources. Also, the production systems are becoming more and more autonomous requiring less operator intervention in every day normal operation. That means using computers for controlling and supervision of the production systems, industrial networks and distributed software architectures [1]. It means also designing application software that is really distributed in the shop floor, taking advantage of the flexibility installed by using programmable equipment.

Non-flat ceramic products are commonly used in our homes and are mainly associated with personal care tasks. The industrial production of those ceramic products poses several problems to industrial automation, namely if robots are to be used. Basically, those problems arise from the characteristics of the ceramic pieces: non-flat objects with high reflective surfaces, very difficult to grasp and handle due to the external configuration, very heavy and fragile, extensive surface sensitive to damage, high demand of quality on surface smoothness, etc. Also, the production setups for this type of products require very high quality and low cycle times, since this is a large scale industry that will only remain competitive if production rates are kept. Another restriction is related with the fact that this industry changes products very frequently, due to fashion tendencies in home decoration, etc. Also, there is an extensive mixture between automatic and human labour production, which is a difficult problem since Human-Machine Interfaces (HMI) are very demanding and a key issue in modern industrial automation systems.

The paper describes a prototype developed to de-palletize non-flat ceramic pieces in the final stage of production, just after they leave the high temperature oven, feeding the final human operated inspection lines. The prototype is installed at ROCA, a Spanish company operating in Portugal, and works with all their models of toilets and bidets.

The rest of the paper is organized as follows: section 2 briefly describes the problem and the defined objectives. Section 3, describes the solution adopted and briefly outlines the main characteristics of the software used for development. Section 4 explores the functionality of the obtained system, taking special attention to the HMI software. Finally conclusions are drawn in section 5.

II. Objectives

The main objective was to build a robotic system that could be used to feed the final inspection lines (Fig. 1). The system should be able to work with pallets composed by 4 levels of ceramic pieces, 8 pieces per level placed in a special order to keep pallet equilibrium, and with levels separated with pieces of hard paper. The rule used to arrange the pieces in the pallet is to place them alternatively one up - one down, starting from the ground level, then swap to one down - one up in the next level (Fig. 2) and keep the procedure in the proceeding levels. Levels are numerated from up to bottom, i.e., level 1 is the top level and level 4 is the bottom level.

Actually pallets are assembled manually by operators at the end of the high temperature oven. This means that the robotic system must be tolerant with possible medium-large palletizing errors, coming from misplaced pieces both in position and orientation, and showing also significant variations from level to level. Another important thing is that pallets are fed into the system by human operators using electro-mechanic pile drivers, which also introduces some variation in the pallet. Sometime in the future AGVs will be use to fulfill the task, reducing considerably the variations introduced and increasing efficiency of the system.
Figure 1 - Basic layout of the system.

Figure 2 - Organization of ceramic pieces in the pallets.
Other important requirements include:

1. Possibility to easily introduce new product models;
2. Possibility to parameterize the operation on each model, so that best performance per model is achieved;
3. Possibility to change model under production without stopping production;
4. Possibility to monitor production using a graphical interface;
5. Obtain medium cycle times of about 12 seconds per piece, which means a new piece in each inspecting line every 24 seconds.

All these objects were addressed and a brief comment about each of them will be given in the next sections.

III. Software

Having these objectives in mind it was decided to operate the system using an external personal computer, using the teach pendant of the robot only for a few special routines not necessary in every day normal operations. Client-server software architecture was adopted. The robot controller software works as a server, exposing to the client a collection of services that constitute its basic functionality. A collection of services was designed to fulfill all the tasks required to the system, so that they could be called from the PC (Fig. 3).

The software architecture used in this work, was presented in detail elsewhere [2,3], and is distributed using a client-server model, based on software components developed to handle equipment functionality. Briefly, when we want to use some kind of equipment from a computer we need to write code and define data structures to handle all its functionality. We can then pack the software into libraries, which are not very easy to distribute being language dependant, or build a software control using one of the several standard architectures available (preferably ActiveX or JAVA). Using a software control means implementing methods and data structures that hide from the user all the tricky parts about how to have things done with some equipment, focusing only on using its functions in an easy way. Beside that, those components are easily integrated into new projects built with programming tools that can act as containers of that type of software controls, i.e., they can be added to new projects in a "visual" way.

IV. System Operation

The system is completely operated using a graphical panel running on the PC, built using the above mentioned ActiveX controls in Visual C++ (Fig. 4). When the system is started, the operator needs only to specify what product model will be used in each pallet, and if first pallets are fully assembled. Sometimes, due to production there are some nonFully assembled pallets on the shop floor, and there is the need to introduce those pallets in the system. To be able to do that, the software allows operator to specify the position and level of the first piece. That is however only possible on the first pallet, because the system resets definitions to the next pallets to avoid accidents, i.e., proceeding pallets are assumed to be fully assembled.

When the operator commands "automatic mode" the robot approaches the selected pallet in direction to the actual piece, searches the piece border using laser sensors placed on the gripper, and fetches the ceramic piece. After that the robot places the piece in the first available inspection line, alternating inspection lines if they are both available, i.e., the robot tries to alternate between them, but if the selected one is not available then the other is used if available. If both inspection lines are occupied the robot waits for the first to become available. Figure 4 shows the interface used by the operator to command the system and monitor production. It shows the commands available, and the on-line production data that enables operators to follow production. All commands and events are logged into a log file, so that production managers can use it for production monitoring, planning, debugging, etc. The system uses also a database, organized in function of the model number, where all the data related to each model is stored. That data includes type of the piece, dimensions, height where the gripper should grab the piece, average position of the first piece of the pallet, height of the pallet, etc. Accessing and updating the database is done in "manual mode", selected in the PC interface. There is a "teaching" option that enables operator to introduce new models and parameterize the database for that model (Fig. 5). When that option is commanded, the robot pre-positions near the pallet and the operator can jog the robot using function keys to the desired position/orientation. Basically the de-palletizing operation is preformed step-by-step and the necessary parameters acquired in the process, inquiring the operator to correct and acknowledge when necessary. The operator is only asked to enter the "model number", and to teach the height and the width of the piece. The rest is automatic. After finishing this routine the model is introduce into the database, and the system can then work with that model number.
Figure 3 – Software architecture used.

Robot Control:
Start/Stop
Program Management

Messages:
Errors and warnings

Database Access

Production data and monitoring.

Figure 4 – Interface used by the operator (labels in Portuguese).
V. Discussion

Figure 5 shows some aspects of the system in operation. The system showed to be very easy to operate, and the interface with operators showed to be efficient and easy to use. Operators adapted quickly and like to use it. One of the most important things about introducing robots into production systems, that share also a very high degree of human labour, is the interface between machines and humans. In some sense the interface should be easy to operate, but it should be designed in a way that there are no sudden moves, i.e., operation must start or resume slowly and take always paths well known by the operator. Predictability is an important thing. Safety is another important thing, since humans tend to adapt quickly and relax safety procedures. Because of that, intrusion into the cell must always act on the emergencies of the system as quickly as possible (using class 5 safety equipments), and should be logged for proper action. We adopted to add a special procedure for after emergency re-start that is time consuming, and requires operator login to resume. In that case, all emergencies are logged with the ID of the operator which prevents operators of being too much confident.

The average cycle time obtained is lower than 12 seconds, which complies with the requirement of one piece on each inspection line every 24 seconds.

Another important thing is the time to production, when a new model is introduced. Adding a model should be done easily by any operator, and should not take too long, so that normal production would not be affected. With the actual system, a new model is introduced and ready to handle by the system in less than 5 minutes, including verification tests used to check if programming was well done.

The ideas presented were implemented on the system using an Ethernet network, connecting an industrial robot to a commanding PC and to the other resources of the shop-floor. The results show that using available technology and distributing functionality over the several components of the system, it is possible to explore the flexibility offered by actual automation equipment. Robots are a special case since they can be used to perform human like tasks and are programmable devices. Nevertheless, they are currently somehow limited, since the interfaces are non-standard and limited in functionality, difficult to use and program for advance remote applications. One way to easily distribute and integrate the code into new applications is by using software components, i.e., using ActiveX components, Java Beans, Corba components, etc. We usually use ActiveX do to the fact that they integrate well with Win32 environments. In fact we built several components for a variety of equipment (PLCs, CCD cameras, force/torque sensors, etc) using this language [2,3,7,10,11,12]. But other technologies could be used; the purpose here is on components and on integration with the environment chosen for operation, not in discussing the possibilities of each technology. Since we use win32, mainly Windows NT and 2000, which is an accepted standard in industry, ActiveX is somehow privileged because it was specially built for those environments and is based on DCOM like the operating system [8,9]. There is currently a discussion among researchers about open source software, as a way to give access to developers and allow implementation of their ideas. This is particularly necessary on research environments, where a good access to resources is needed. The problem is real and somewhat urgent, since many features currently common on laboratories didn't reach industry yet due to system interface limitations (robot control systems are closed and with deficient software interfaces). Manufacturers should provide powerful APIs to enable user access to system resources, tailoring its behavior, from a remote computer. And that is a good business decision, because it enables third-party solutions for high demanding and non-traditional applications, which cannot be handled properly by general corporate products. That type of openness will be more important than open source, if it means support and full access to the robotic system (using currently available and accepted standards) by advanced users, which will contribute to the dissemination of the technology to non-traditional, and SME based applications. These types of applications are usually very tricky, very demanding on computer control and support, special sensors [11,12], etc., but not sufficiently important in the number of robots sold to motivate the interest of the robot manufacturers. That is the case of the example presented in this paper.
VI. Conclusions

In this paper an industrial robotic system designed for handling non-flat ceramic products was presented and briefly explained. In the process our approach to design software to this type of systems was introduced and some aspects discussed. Discussion about advantages of this type of approach was briefly outlined, along with recommendations that manufacturers of these types of automation equipments should follow in the near future.

The obtained system proved to be easy to use, efficient and integrated well in the existing production setup, which is fundamental in this type of environments with an extensive mixture between automatic and human labour.

VII. References


